Bidirectional PageRank Estimation: From Average-Case to Worst-Case

Peter Lofgren[1], Siddhartha Banerjee[2], and Ashish Goel[1]

[1] Stanford University, Stanford CA 94305, USA
[2] Cornell University, Ithaca NY 14850, USA

Abstract. We present a new algorithm for estimating the Personalized PageRank (PPR) between a source and target node on undirected graphs, with sublinear running-time guarantees over the worst-case choice of source and target nodes. Our work builds on a recent line of work on bidirectional estimators for PPR, which obtained sublinear running-time guarantees but in an average-case sense, for a uniformly random choice of target node. Crucially, we show how the reversibility of random walks on undirected networks can be exploited to convert average-case to worst-case guarantees. While past bidirectional methods combine forward random walks with reverse local pushes, our algorithm combines forward local pushes with reverse random walks. We also discuss how to modify our methods to estimate random-walk probabilities for any length distribution, thereby obtaining fast algorithms for estimating general graph diffusions, including the heat kernel, on undirected networks.

1 Introduction

Ever since their introduction in the seminal work of Page et al. [23], PageRank and Personalized PageRank (PPR) have become some of the most important and widely used network centrality metrics (a recent survey [13] lists several examples). At a high level, for any graph G, given 'teleport' probability α and a 'personalization distribution' σ over the nodes of G, PPR models the importance of every node from the point of view of σ in terms of the stationary probabilities of 'short' random walks that periodically restart from σ with probability α. It can be defined recursively as giving importance α to σ, and in addition giving every node importance based on the importance of its in-neighbors.

Formally, given normalized adjacency matrix $W = D^{-1}A$, the Personalized PageRank vector π_σ with respect to source distribution σ is the solution to

$$\pi_\sigma = \alpha\sigma + (1-\alpha)\pi_\sigma W. \qquad (1)$$

An equivalent definition is in terms of the terminal node of a random-walk starting from σ. Let $\{X_0, X_1, X_2, \ldots\}$ be a random-walk starting from $X_0 \sim \sigma$, and $L \sim Geometric(\alpha)$. Then the PPR of any node t is given by [4]:

$$\pi_\sigma(t) = \mathbb{P}[X_L = t] \qquad (2)$$

The equivalence of these definitions can be seen using a power series expansion.

In this work, we focus on developing PPR-estimators with *worst-case sublinear guarantees* for *undirected graphs*. Apart from their technical importance, our results are are of practical relevance as several large-scale applications of PPR are based on undirected networks. For example, Facebook (which is an undirected social network) used Personalized PageRank for friend recommendation [5]. The social network Twitter is directed, but Twitter's friend recommendation algorithm (Who to Follow) [16] uses an algorithm called personalized SALSA [19,6], which first converts the directed network into an expanded undirected graph [3], and then computes PPR on this new graph. Random walks have also been used for collaborative filtering by the YouTube team [7] (on the undirected user-item bipartite graph), to predict future items a user will view. Applications like this motivate fast algorithms for PPR estimation on undirected graphs.

Equations (1) and (2) suggest two natural estimation algorithms for PPR – via linear-algebraic iterative techniques, and using Monte Carlo. The linear algebraic characterization of PageRank in Eqn. (1) suggests the use of power iteration (or other localized iterations; cf Section 1.2 for details), while Eqn. (2) is the basis for a Monte-Carlo algorithm, wherein we estimate $\pi_\sigma[t]$ by sampling independent L-step paths, each starting from a random state sampled from σ. For studying PageRank estimation algorithms, smaller probabilities are more difficult to estimate than large ones, so a natural parametrization is in terms of the minimum PageRank we want to detect. Formally, given any source σ, target node $t \in V$ and a desired minimum probability threshold δ, we want algorithms that give accurate estimates whenever $\pi_\sigma[t] \geq \delta$. Improved algorithms are motivated by the slow convergence of these algorithms: *both Monte Carlo and linear algebraic techniques have a running time of $\Omega(1/\delta)$* for PageRank estimation. Furthermore this is true not only for worst case choices of target state t, but on average Monte-Carlo requires $\Omega(1/\delta)$ time to estimate a probability of size δ. Power iteration takes $\Theta(m)$ time, where m is the number of edges, and the work [21] shows empirically that the local version of power-iteration scales with $1/\delta$ for $\delta > 1/m$.

In a recent line of work, linear-algebraic and Monte-Carlo techniques were combined to develop new *bidirectional PageRank* estimators `FAST-PPR` [22] and `Bidirectional-PPR` [20], which gave the first significant improvement in the running-time of PageRank estimation since the development of Monte-Carlo techniques. Given an arbitrary source distribution σ and a *uniform random target node* t, these estimators were shown to return an accurate PageRank estimate with an *average* running-time of $\tilde{O}\left(\sqrt{\overline{d}/\delta}\right)$, where $\overline{d} = m/n$ is the average degree of the graph. Given $\tilde{O}\left(n\sqrt{\overline{d}/\delta}\right)$ precomputation and storage,

---

[3] Specifically, for each node u in the original graph, SALSA creates two virtual nodes, a "consumer-node" u' and a "producer-node" u'', which are linked by an undirected edge. Any directed edge (u, v) is then converted into an undirected edge (u', v'') from u's consumer node to v's producer node.

the authors prove worst case guarantees for this bidirectional estimator but in practice that is a large precomputation requirement. This raised the challenge of designing an algorithm with similar running-time guarantees over a *worst-case* choice of target node t. Inspired by the bidirectional estimators in [22,20], we propose a new PageRank estimator for *undirected graphs* with *worst-case* running time guarantees.

1.1 Our Contribution

We present the first estimator for personalized PageRank with sublinear running time in the worst case on undirected graphs. We formally present our `Undirected-BiPPR` algorithm in Section 2, and prove that it has the following accuracy and running-time guarantees:

Result 1 (See Theorem 1 in Section 2) *Given any undirected graph G, teleport probability α, source node s, target node t, threshold δ and relative error ϵ, the* `Undirected-BiPPR` *estimator (Algorithm 2) returns an unbiased estimate $\widehat{\pi}_s[t]$ for $\pi_s[t]$, which, with probability greater than $1 - p_{fail}$, satisfies:*

$$|\widehat{\pi}_s[t] - \pi_s[t]| < \max\left\{\epsilon\pi_s[t], 2e\delta\right\}.$$

Result 2 (See Theorem 2 in Section 2) *Let any undirected graph G, teleport probability α, threshold δ and desired relative error ϵ be given. For any source, target pair (s,t), the* `Undirected-BiPPR` *algorithm has a running-time of $O\left(\frac{\sqrt{\ln(1/p_{fail})}}{\epsilon}\sqrt{\frac{d_t}{\delta}}\right)$, where d_t is the degree of the target node t.*

In personalization applications, we are often only interested in personalized importance scores if they are greater than global importance scores, so it is natural to set δ based on the global importance of t. Assuming G is connected, in the limit $\alpha \to 0$, the PPR vector for any start node s converges to the stationary distribution of infinite-length random-walks on G – that is $\lim_{\alpha\to 0}\pi_s[t] = d_t/m$. This suggests that a natural PPR significance-test is to check whether $\pi_s(t) \geq d_t/m$. To this end, we have the following corollary:

Result 3 (See Corollary 1 in Section 2) *For any graph G and any (s,t) pair such that $\pi_s(t) \geq \frac{d_t}{m}$, then with high probability [4],* `Undirected-BiPPR` *returns an estimate $\pi_s(t)$ with relative error ϵ with a* worst-case *running-time of $O\left(\sqrt{m}\log n/\epsilon\right)$.*

Finally, in Section 3, using ideas from [8], we extend our technique to estimating more general random-walk transition-probabilities on undirected graphs, including graph diffusions and the heat kernel [11,18].

---

[4] Following convention, we use w.h.p. to mean with probability greater than $1 - \frac{1}{n}$.

1.2 Existing Approaches for PageRank Estimation

We first summarize the existing methods for PageRank estimation:

Monte Carlo Methods: A standard method [4,9] for estimating $\pi_\sigma[t]$ is by using the terminal node of independently generated random walks of length $L \sim Geometric(\alpha)$ starting from a random node sampled from σ. Simple concentration arguments show that we need $\widetilde{\Theta}(1/\delta)$ samples to get an accurate estimate of $\pi_\sigma[t]$, irrespective of the choice of t and graph G.

Linear-Algebraic Iterations: Since the PageRank vector is the stationary distribution of a Markov chain, it can also be estimated via forward or reverse power iterations. A direct power iteration is often infeasible for large graphs; in such cases, it is preferable to use localized power iterations [2,1]. These local-update methods can also be used for other transition probability estimation problems such as heat kernel estimation [18]. Local update algorithms are often fast in practice, as unlike full power iteration methods they exploit the local structure of the chain. However even in sparse Markov chains and for a large fraction of target states, their running time can be $\Omega(1/\delta)$. For example, consider a random walk on a random d-regular graph and let $\delta = o(1/n)$. Then for $\ell \sim \log_d(1/\delta)$, verifying $\pi_{\underline{\mathbf{e}}_s}[t] > \delta$ is equivalent to uncovering the entire $\log_d(1/\delta)$ neighborhood of s. However since a large random d-regular graph is (w.h.p.) an expander, this neighborhood has $\Omega(1/\delta)$ distinct nodes.

Bidirectional Techniques: Bidirectional methods are based on simultaneously working forward from the source node s and backward from the target node t in order to improve the running-time. One example of such a bidirectional technique is the use of *colliding random-walks* to estimate length-2ℓ random-walk transition probabilities in *regular undirected graphs* [14,17] – the main idea here is to exploit the reversibility by using two independent random walks of length ℓ starting from s and t respectively, and detecting if they collide. This results in reducing the number of walks required by a square-root factor, based on an argument similar to the birthday-paradox.

The `FAST-PPR` algorithm of Lofgren et al. [22] was the first bidirectional algorithm for estimating PPR in general graphs; this was subsequently refined and improved by the `Bidirectional-PPR` algorithm [20], and also generalized to other Markov chain estimation problems [8]. These algorithms are based on using a reverse local-update iteration from the target t (adapted from Andersen et al. [1]) to smear the mass over a larger *target set*, and then using random-walks from the source s to detect this target set. From a theoretical perspective, a significant breakthrough was in showing that for arbitrary choice of source node s these bidirectional algorithms achieved an *average* running-time of $\tilde{O}(\sqrt{d/\delta})$ over uniform-random choice of target node t – in contrast, both local-update and Monte Carlo has a running-time of $\Omega(1/\delta)$ for uniform-random targets. More recently, [10] showed that a similar bidirectional technique achieved a sublinear query-complexity for global PageRank computation, under a modified query model, in which all neighbors of a given node could be found in $O(1)$ time.

2 PageRank Estimation in Undirected Graphs

We now present our new bidirectional algorithm for PageRank estimation in undirected graphs.

2.1 Preliminaries

We consider an undirected graph $G(V, E)$, with n nodes and m edges. For ease of notation, we henceforth consider unweighted graphs, and focus on the simple case where $\sigma = \underline{\mathbf{e}}_s$ for some single node s. We note however that all our results extend to weighted graphs and any source distribution σ in a straightforward manner.

2.2 A Symmetry for PPR in Undirected Graphs

The `Undirected-BiPPR` Algorithm critically depends on an underlying *reversibility property* exhibited by PPR vectors in undirected graphs. This property, stated before in several earlier works [3,15], is a direct consequence of the reversibility of random walks on undirected graphs. To keep our presentation self-contained, we present this property, along with a simple probabilistic proof, in the form of the following lemma:

Lemma 1. *Given any undirected graph G, for any teleport probability $\alpha \in (0,1)$ and for any node-pair $(s,t) \in V^2$, we have:*

$$\pi_s[t] = \frac{d_t}{d_s} \pi_t[s].$$

Proof. For path $P = \{s, v_1, v_2, \ldots, v_k, t\}$ in G, we denote its length as $\ell(P)$ (here $\ell(P) = k + 1$), and define its reverse path to be $\overline{P} = \{t, v_k, \ldots, v_2, v_1, s\}$ – note that $\ell(P) = \ell(\overline{P})$. Moreover, we know that a random-walk starting from s traverses path P with probability $\mathbb{P}[P] = \frac{1}{d_s} \cdot \frac{1}{d_{v_1}} \cdot \ldots \cdot \frac{1}{d_{v_k}}$, and thus, it is easy to see that we have:

$$\mathbb{P}[P] \cdot d_s = \mathbb{P}[\overline{P}] \cdot d_t \tag{3}$$

Now let $\mathcal{P}_{st}$ denote the set of paths in G starting at s and terminating at t. Then we can re-write Eqn. (2) as:

$$\pi_s[t] = \prod_{P \in \mathcal{P}_{st}} \alpha(1-\alpha)^{\ell(P)} \mathbb{P}[P] = \prod_{\overline{P} \in \mathcal{P}_{ts}} \alpha(1-\alpha)^{\ell(\overline{P})} \mathbb{P}[\overline{P}] = \frac{d_t}{d_s} \pi_t[s] \quad \square$$

2.3 The `Undirected-BiPPR` Algorithm

At a high level, the `Undirected-BiPPR` algorithm has two components:

– **Forward-work**: Starting from source s, we first use a forward local-update algorithm, the $\texttt{ApproximatePageRank}(G, \alpha, s, r_{\max})$ algorithm of Andersen et al. [2] (shown here as Algorithm 1). This procedure begins by placing one unit of "residual" probability-mass on s, then repeatedly selecting some node u, converting an α-fraction of the residual mass at u into probability mass, and pushing the remaining residual mass to u's neighbors. For any node u, it returns an estimate $p_s[u]$ of its PPR $\pi_s[u]$ from s as well as a residual $r_s[u]$ which represents un-pushed mass at u.

– **Reverse-work**: We next sample random walks of length $L \sim Geometric(\alpha)$ starting from t, and use the residual at the terminal nodes of these walks to compute our desired PPR estimate. Our use of random walks backwards from t depends critically on the symmetry in undirected graphs presented in Lemma 1.

Note that this is in contrast to $\texttt{FAST-PPR}$ and $\texttt{Bidirectional-PPR}$, which performs the local-update step in reverse from the target t, and generates random-walks forwards from the source s.

Algorithm 1 $\texttt{ApproximatePageRank}(G, \alpha, s, r_{\max})$ [2]

Inputs: graph G, teleport probability α, start node s, maximum residual $r_{\max}$
 1: Initialize (sparse) estimate-vector $p_s = \mathbf{0}$ and (sparse) residual-vector $r_s = e_s$ (i.e. $r_s[v] = 1$ if $v = s$; else 0)
 2: **while** $\exists u \in V$ s.t. $\frac{r_s[u]}{d_u} > r_{\max}$ **do**
 3:   **for** $v \in \mathcal{N}[u]$ **do**
 4:     $r_s[v] \mathrel{+}= (1 - \alpha)r_s[u]/d_u$
 5:   **end for**
 6:   $p_s[u] \mathrel{+}= \alpha r_s[u]$
 7:   $r_s[u] = 0$
 8: **end while**
 9: **return** (p_s, r_s)

In more detail, our algorithm will choose a maximum residual parameter $r_{\max}$, and apply the local push operation in Algorithm 1 until for all v, $r_s[v]/d_v < r_{\max}$. Andersen et al. [2] prove that their local-push operation preserves the following invariant for vectors (p_s, r_s):

$$\pi_s[t] = p_s[t] + \sum_{v \in V} r_s[v]\pi_v[t], \qquad \forall\, t \in V. \tag{4}$$

Since we ensure that $\forall v, r_s[v]/d_v < r_{\max}$, it is natural at this point to use the symmetry Lemma 1 and re-write this as:

$$\pi_s[t] = p_s[t] + d_t \sum_{v \in V} \frac{r_s[v]}{d_v}\pi_t[v].$$

Now using the fact that $\sum_t \pi_v[t] = n\pi[t]$ get that $\forall\, t \in V$, $|\pi_s[t] - p_s[t]| \le r_{\max}d_t n\pi[t]$.

However, we can get a more accurate estimate by using the residuals. The key idea of our algorithm is to re-interpret this as an expectation:

$$\pi_s[t] = p_s[t] + d_t \mathbb{E}_{V \sim \pi_t} \left[\frac{r_s[v]}{d_V} \right]. \tag{5}$$

We estimate the expectation using standard Monte-Carlo. Let $V_i \sim \pi_t$ and $X_i = r_s(V_i) d_t / d_{V_i}$, so we have $\pi_s[t] = p_s[t] + \mathbb{E}[X]$. Moreover, each sample X_i is bounded by $d_t r_{\max}$ (this is the stopping condition for `ApproximatePageRank`), which allows us to efficiently estimate its expectation. To this end, we generate w random walks, where

$$w = \frac{c}{\epsilon^2} \frac{r_{\max}}{\delta / d_t}.$$

The choice of c is specified in Theorem 1. Finally, we return the estimate:

$$\widehat{\pi}_s[t] = p_t[s] + \frac{1}{w} \sum_{i=1}^{w} X_i.$$

The complete pseudocode is given in Algorithm 2.

Algorithm 2 `Undirected-BiPPR`(s, t, δ)

Inputs: graph G, teleport probability α, start node s, target node t, minimum probability δ, accuracy parameter $c = 3 \ln (2/p_{\text{fail}})$ (cf. Theorem 1)

1: $(p_s, r_s) = $ `ApproximatePageRank`$(s, r_{\max})$
2: Set number of walks $w = c d_t r_{\max} / (\epsilon^2 \delta)$
3: **for** index $i \in [w]$ **do**
4:     Sample a random walk starting from t, stopping after each step with probability α; let V_i be the endpoint
5:     Set $X_i = r_s(V_i)/d_{V_i}$
6: **end for**
7: **return** $\widehat{\pi}_s[t] = p_s[t] + (1/w) \sum_{i \in [w]} X_i$

2.4   Analyzing the Performance of `Undirected-BiPPR`

Accuracy Analysis: We first prove that `Undirected-BiPPR` returns an unbiased estimate with the desired accuracy:

Theorem 1. *In an undirected graph G, for any source node s, minimum threshold δ, maximum residual r_{max}, relative error ϵ, and failure probability p_{fail}, Algorithm 2 outputs an estimate $\widehat{\pi}_s[t]$ such that with probability at least $1 - p_{fail}$ we have:* $|\pi_s[t] - \hat{\pi}_s[t]| \leq \max\{\epsilon \pi_s[t], 2e\delta\}$.

The proof follows a similar outline as the proof of Theorem 1 in [20]. For completeness, we sketch the proof here:

Proof. As stated in Algorithm 2, we average over $w = cd_t r_{\max}/\epsilon^2 \delta$ walks, where c is a parameter we choose later. Each walk is of length $Geometric(\alpha)$, and we denote V_i as the last node visited by the ith walk; note that $V_i \sim \pi_t$. As defined above, let $X_i = r_s(V_i)d_t/d_{V_i}$; the estimate returned by `Undirected-BiPPR` is:

$$\widehat{\pi}_s[t] = p_t[s] + \frac{1}{w} \sum_{i=1}^{w} X_i.$$

First, from Eqn. (5), we have that $\mathbb{E}[\widehat{\pi}_s[t]] = \pi_s[t]$. Also, `ApproximatePageRank` guarantees that for all v, $r_s[v] < d_v r_{\max}$, and so each X_i is bounded in $[0, d_t r_{\max}]$; for convenience, we rescale X_i by defining $Y_i = \frac{1}{d_t r_{\max}} X_i$.

We now show concentration of the estimates via the following Chernoff bounds (see Theorem 1.1 in [12]):

1. $\mathbb{P}[|Y - \mathbb{E}[Y]| > \epsilon \mathbb{E}[Y]] < 2 \exp(-\frac{\epsilon^2}{3}\mathbb{E}[Y])$
2. For any $b > 2e\mathbb{E}[Y], \mathbb{P}[Y > b] \leq 2^{-b}$

We perform a case analysis based on whether $\mathbb{E}[X_i] \geq \delta$ or $\mathbb{E}[X_i] < \delta$. First, if $\mathbb{E}[X_i] \geq \delta$, then we have $\mathbb{E}[Y] = \frac{w}{d_t r_{\max}}\mathbb{E}[X_i] = \frac{c}{\epsilon^2 \delta}\mathbb{E}[X_i] \geq \frac{c}{\epsilon^2}$, and thus:

$$\mathbb{P}\left[|\widehat{\pi}_s[t] - \pi_s[t]| > \epsilon \pi_s[t]\right] \leq \mathbb{P}\left[\left|\bar{X} - \mathbb{E}[X_i]\right| > \epsilon \mathbb{E}[X_i]\right] = \mathbb{P}\left[|Y - \mathbb{E}[Y]| > \epsilon \mathbb{E}[Y]\right]$$

$$\leq 2 \exp\left(-\frac{\epsilon^2}{3}\mathbb{E}[Y]\right) \leq 2 \exp\left(-\frac{c}{3}\right) \leq p_{\text{fail}},$$

where the last line holds as long as we choose $c \geq 3 \ln(2/p_{\text{fail}})$.

Suppose alternatively that $\mathbb{E}[X_i] < \delta$. Then:

$$\mathbb{P}[|\hat{\pi}_s[t] - \pi_s[t]| > 2e\delta] = \mathbb{P}[|\bar{X} - \mathbb{E}[X_i]| > 2e\delta] = \mathbb{P}\left[|Y - \mathbb{E}[Y]| > \frac{w}{d_t r_{\max}} 2e\delta\right]$$

$$\leq \mathbb{P}\left[Y > \frac{w}{d_t r_{\max}} 2e\delta\right].$$

At this point we set $b = 2e\delta w/d_t r_{\max} = 2ec/\epsilon^2$ and apply the second Chernoff bound. Note that $\mathbb{E}[Y] = c\mathbb{E}[X_i]/\epsilon^2 \delta < c/\epsilon^2$, and hence we satisfy $b > 2e\mathbb{E}[Y]$. We conclude that:

$$\mathbb{P}[|\hat{\pi}_s[t] - \pi_s[t]| > 2e\delta] \leq 2^{-b} \leq p_{\text{fail}}$$

as long as we choose c such that $c \geq \frac{\epsilon^2}{2e} \log_2 \frac{1}{p_{\text{fail}}}$. The proof is completed by combining both cases and choosing $c = 3 \ln(2/p_{\text{fail}})$. $\quad \square$

Running Time Analysis: The more interesting analysis is that of the running-time of `Undirected-BiPPR` – we now prove a worst-case running-time bound:

Theorem 2. *In an undirected graph, for any source node (or distribution) s, target t with degree d_t, threshold δ, maximum residual r_{max}, relative error ϵ, and failure probability p_{fail}, `Undirected-BiPPR` has a worst-case running-time of:*

$$O\left(\frac{\sqrt{\log \frac{1}{p_{fail}}}}{\epsilon} \sqrt{\frac{d_t}{\delta}}\right).$$

Before proving this result, we first state and prove a crucial lemma from [2]:

Lemma 2 (Lemma 2 **in [2]).** *Let T be the total number of push operations performed by* `ApproximatePageRank`, *and let d_k be the degree of the vertex involved in the kth push. Then:*

$$\sum_{k=1}^{T} d_k \leq \frac{1}{\alpha r_{max}}$$

Proof. Let v_k be the vertex pushed in the kth step – then by definition, we have that $r_s(v_k) > r_{max}d_k$. Now after the local-push operation, the sum residual $||r_s||_1$ decreases by at least $\alpha r_{max}d_k$. However, we started with $||r_s||_1 = 1$, and thus we have $\sum_{k=1}^{T} \alpha r_{max}d_k \leq 1$. $\square$

Note also that the amount of work done while pushing from a node v is d_v.

Proof (of Theorem 2). As proven in Lemma 2, the push forward step takes total time $O\left(1/\alpha r_{max}\right)$ in the *worst-case*. The random walks take $O(w) = O\left(\frac{1}{\epsilon^2}\frac{r_{max}}{\delta/d_t}\right)$ time. Thus our total time is

$$O\left(\frac{1}{r_{max}} + \frac{\ln\frac{1}{p_{fail}}}{\epsilon^2}\frac{r_{max}}{\delta/d_t}\right).$$

Balancing this by choosing $r_{max} = \frac{\epsilon}{\sqrt{\ln\frac{1}{p_{fail}}}}\sqrt{\delta/d_t}$, we get total running-time:

$$O\left(\frac{\sqrt{\ln\frac{1}{p_{fail}}}}{\epsilon}\sqrt{\frac{d_t}{\delta}}\right). \qquad \square$$

We can get a cleaner worst-case running time bound if we make a natural assumption on $\pi_s[t]$. In an undirected graph, if we let $\alpha = 0$ and take infinitely long walks, the stationary probability of being at any node t is $\frac{d_t}{m}$. Thus if $\pi_s[t] < \frac{d_t}{m}$, then s actually has a lower PPR to t than the non-personalized stationary probability of t, so it is natural to say t is not significant for s. If we set a significance threshold of $\delta = \frac{d_t}{m}$, and apply the previous theorem, we immediately get the following:

Corollary 1. *If $\pi_s[t] \geq \frac{d_t}{m}$, we can estimate $\pi_s[t]$ within relative error ϵ with probability greater than $1 - \frac{1}{n}$ in worst-case time:*

$$O\left(\frac{\log n}{\epsilon}\sqrt{m}\right).$$

In contrast, the running time for `Monte-Carlo` to achieve the same accuracy guarantee is $O\left(\frac{1}{\delta}\frac{\log(1/p_{fail})}{\alpha\epsilon^2}\right)$, and the running time for `ApproximatePageRank` is $O\left(\frac{\bar{d}}{\delta\alpha}\right)$. The `FAST-PPR` algorithm of [22] has an *average case* running time of

$O\left(\frac{1}{\alpha\epsilon^2}\sqrt{\frac{\bar{d}}{\delta}}\sqrt{\frac{\log(1/p_{\text{fail}})\log(1/\delta)}{\log(1/(1-\alpha))}}\right)$ for uniformly chosen targets, but has no clean worst-case running time bound because its running time depends on the degree of nodes pushed from in the linear-algebraic part of the algorithm.

3 Extension to Graph Diffusions

PageRank and Personalized PageRank are a special case of a more general set of network-centrality metrics referred to as *graph diffusions* [11,18]. In a a graph diffusion we assign a weight α_i to walks of length i. The score is then is a polynomial function of the random-walk transition probabilities of the form:

$$f(W, \sigma) := \sum_{i=0}^{\infty} \alpha_i \left(\sigma W^i\right),$$

where $\alpha_i \geq 0, \sum_i \alpha_i = 1$. To see that PageRank has this form, we can expand Eqn. (1) via a Taylor series to get:

$$\pi_\sigma = \sum_{i=1}^{\infty} \alpha(1-\alpha)^i \left(\sigma W^i\right)$$

Another important graph diffusion is the *heat kernel* h_σ, which corresponds to the scaled matrix exponent of $(I - W)^{-1}$:

$$h_{\sigma,\gamma} = e^{-\gamma(I-W)^{-1}} = \sum_{i=1}^{\infty} \frac{e^{-\gamma}\gamma^i}{i!} \left(\sigma W^i\right)$$

In [8], Banerjee and Lofgren extended `Bidirectional-PPR` to get bidirectional estimators for graph diffusions and other general Markov chain transition-probability estimation problems. These algorithms inherited similar performance guarantees to `Bidirectional-PPR` – in particular, they had good expected running-time bounds for uniform-random choice of target node t. We now briefly discuss how we can modify `Undirected-BiPPR` to get an estimator for graph diffusions in undirected graphs with worst-case running-time bounds.

First, we observe that Lemma 1 extends to all graph diffusions, as follows:

Corollary 2. *Let any undirected graph G with random-walk matrix W, and any set of non-negative length weights $(\alpha_i)_{i=0}^{\infty}$ with $\sum \alpha_i = 1$ be given. Define $f(W, \sigma) = \sum_{i=0}^{\infty} \alpha_i \left(\sigma W^i\right)$. Then for any node-pair $(s,t) \in V^2$, we have:*

$$f\left(W, \underline{\mathbf{e}}_s\right) = \frac{d_t}{d_s} f\left(W, \underline{\mathbf{e}}_t\right).$$

As before, the above result is stated for unweighted graphs, but it also extends to random-walks on weighted undirected graphs, if we define $d_i = \sum_j w_{ij}$.

Next, observe that for any graph diffusion $f(\cdot)$, the truncated sum $f^{\ell_{\max}} = \sum_{i=0}^{\ell_{\max}} \alpha_i \left(\pi_\sigma^T P^i\right)$ obeys: $||f - f^{\ell_{\max}}||_\infty \leq \sum_{\ell_{\max}+1}^{\infty} \alpha_k$ Thus a guarantee on an

estimate for the truncated sum directly translates to a guarantee on the estimate for the diffusion.

The main idea in [8] is to generalize the bidirectional estimators for PageRank to estimating *multi-step transitions probabilities* (for short, MSTP). Given a source node s, a target node t, and length $\ell \leq \ell_{\max}$, we define:

$$p_s^\ell[t] = \mathbb{P}[\text{Random-walk of length } \ell \text{ starting from } s \text{ terminates at } t]$$

Note from Corollary 2, we have for any pair (s,t) and any ℓ, $p_s^\ell[t]d_s = p_t^\ell[s]d_t$.

Now in order to develop a bidirectional estimator for $p_s^\ell[t]$, we need to define a local-update step similar to `ApproximatePageRank`. For this, we can modify the `REVERSE-PUSH` algorithm from [8], as follows.

Similar to `ApproximatePageRank`, given a source node s and maximum length $\ell_{\max}$, we associate with each length $\ell \leq \ell_{\max}$ an estimate vector q_s^ℓ and a residual vector r_s^ℓ. These are updated via the following `ApproximateMSTP` algorithm:

Algorithm 3 ApproximateMSTP$(G, s, \ell_{\max}, r_{\max})$

Inputs: Graph G, source s, maximum steps $\ell_{\max}$, maximum residual $r_{\max}$
1: Initialize: Estimate-vectors $q_s^k = \underline{0}$, $\forall k \in \{0, 1, 2, \ldots, \ell_{\max}\}$,
               Residual-vectors $r_s^0 = \underline{\mathbf{e}}_s$ and $r_s^k = \underline{0}$, $\forall k \in \{1, 2, 3, \ldots, \ell_{\max}\}$
2: **for** $i \in \{0, 1, \ldots, \ell_{\max}\}$ **do**
3:    **while** $\exists v \in \mathcal{S}$ $s.t.$ $r_t^i[v]/d_v > r_{\max}$ **do**
4:       **for** $w \in \mathcal{N}(v)$ **do**
5:          $r_s^{i+1}[w] \mathrel{+}= r_s^i[v]/d_v$
6:       **end for**
7:       $q_s^i[v] \mathrel{+}= r_s^i[v]$
8:       $r_s^i[v] = 0$
9:    **end while**
10: **end for**
11: **return** $\{q_s^\ell, r_s^\ell\}_{\ell=0}^{\ell_{\max}}$

The main observation now is that for any source s, target t, and length ℓ, after executing the `ApproximateMSTP` algorithm, the vectors $\{q_s^\ell, r_s^\ell\}_{\ell=0}^{\ell_{\max}}$ satisfy the following invariant (via a similar argument as in [8], Lemma 1):

$$p_s^\ell[t] = q_s^\ell[t] + \sum_{k=0}^{\ell} \sum_{v \in V} r_s^k[v] p_v^{\ell-k}[t] = q_s^\ell[t] + d_t \sum_{k=0}^{\ell} \sum_{v \in V} \frac{r_s^k[v]}{d_v} p_t^{\ell-k}[v]$$

As before, note now that the last term can be written as an expectation over random-walks originating from t. The remaining algorithm, accuracy analysis, and runtime analysis follow the same lines as those in Section 2.

4 Lower Bound

In [22], the authors prove an average case lower bound for PPR-Estimation. In particular they prove that there exists a family of undirected 3-regular graphs

for which any algorithm that can distinguish between pairs (s, t) with $\pi_s[t] > \delta$ and pairs (s, t) with $\pi_s[t] < \frac{\delta}{2}$ (distinguishing correctly with constant probability 8/9), must access $\Omega(1/\sqrt{\delta})$ edges of the graph. Since the algorithms in [22,20] solve this problem in time $O\left(\sqrt{\overline{d}/\delta}\right)$, where $\overline{d}$ is the average degree of the given graph, there remains a $\sqrt{d}$ gap between the lower bound and the best algorithm for the average case. For the worst case (possibly parameterized by some property of the graph or target node), the authors are unaware of any lower bound stronger than this average case bound, and an interesting open question is to prove a lower bound for the worst case.

5 Conclusion

We have developed `Undirected-BiPPR`, a new bidirectional PPR-estimator for undirected graphs, which for any (s, t) pair such that $\pi_s[t] > d_t/m$, returns an estimate with ϵ relative-error in worst-case running time of $O(\sqrt{m}/\epsilon)$. This thus extends the average-case running-time improvements achieved in [22,20] to worst-case bounds on undirected graphs, using the reversibility of random-walks on undirected graphs. Whether such worst-case running-time results extend to general graphs, or if PageRank computation is fundamentally easier on undirected graphs as opposed to directed graphs, remains an open question.

6 Acknowledgments

References

1. R. Andersen, C. Borgs, J. Chayes, J. Hopcraft, V. S. Mirrokni, and S.-H. Teng. Local computation of pagerank contributions. In *Algorithms and Models for the Web-Graph*. Springer, 2007.
2. R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, 2006.
3. K. Avrachenkov, P. Gonçalves, and M. Sokol. On the choice of kernel and labelled data in semi-supervised learning methods. In *Algorithms and Models for the Web Graph*, pages 56–67. Springer, 2013.
4. K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova. Monte carlo methods in pagerank computation: When one iteration is sufficient. *SIAM Journal on Numerical Analysis*, 2007.
5. L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011.

6. B. Bahmani, A. Chowdhury, and A. Goel. Fast incremental and personalized pagerank. *Proceedings of the VLDB Endowment*, 4(3):173–184, 2010.

7. S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008.

8. S. Banerjee and P. Lofgren. Fast bidirectional probability estimation in markov models. In *NIPS*, 2015.

9. C. Borgs, M. Brautbar, J. Chayes, and S.-H. Teng. A sublinear time algorithm for pagerank computations. In *Algorithms and Models for the Web Graph*, pages 41–53. Springer, 2012.

10. M. Bressan, E. Peserico, and L. Pretto. Approximating pagerank locally with sublinear query complexity. *arXiv preprint arXiv:1404.1864*, 2014.

11. F. Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 2007.

12. D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

13. D. F. Gleich. PageRank beyond the web. *arXiv*, cs.SI:1407.5107, 2014. Accepted for publication in SIAM Review.

14. O. Goldreich and D. Ron. On testing expansion in bounded-degree graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*. Springer, 2011.

15. V. Grolmusz. A note on the pagerank of undirected graphs. *Information Processing Letters*, 2015.

16. P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. Wtf: The who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 505–514. International World Wide Web Conferences Steering Committee, 2013.

17. S. Kale, Y. Peres, and C. Seshadhri. Noise tolerance of expanders and sublinear expander reconstruction. In *IEEE FOCS'08*. IEEE, 2008.

18. K. Kloster and D. F. Gleich. Heat kernel based community detection. In *ACM SIGKDD'14*, 2014.

19. R. Lempel and S. Moran. The stochastic approach for link-structure analysis (salsa) and the tkc effect. *Computer Networks*, 33(1):387–401, 2000.

20. P. Lofgren, S. Banerjee, and A. Goel. Personalized pagerank estimation and search: A bidirectional approach. In *WSDM*, 2016.

21. P. Lofgren and A. Goel. Personalized pagerank to a target node. *arXiv preprint arXiv:1304.4658*, 2013.

22. P. A. Lofgren, S. Banerjee, A. Goel, and C. Seshadhri. FAST-PPR: Scaling Personalized PageRank estimation for large graphs. In *ACM SIGKDD'14*. ACM, 2014.

23. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.